WEST

Help | Logout | Interrupt

Main Menu | Search Form | Posting Counts | Show S Numbers | Edit S Numbers | Preferences | Cases

Search Results -

Terms	Documents
L4 same (operation code or opcode)	13

Database:
US Patents Full-Text Database
US Pre-Grant Publication Full-Text Database
JPO Abstracts Database
EPO Abstracts Database
Derwent World Patents Index
IBM Technical Disclosure Bulletins

Search: L5

Refine Search

Recall Text | Clear

Search History

DATE: Wednesday, June 11, 2003 Printable Copy Create Case

Set Name side by side	Query	Hit Count	Set Name result set
	DB=USPT; PLUR=YES; OP=ADJ		
L5	L4 same (operation code or opcode)	13	L5
L4	L3 same l2	174	L4
L3	select$3 near3 (unit or section or block or logic or circuit$3 or means)	192201	L3
L2	L1 same (buffer$3 or register or memory)	2323	L2
L1	(multiple or plurality or more than one or various or first) near2 operand	3582	L1

END OF SEARCH HISTORY

WEST

Generate Collection | Print

Search Results - Record(s) 1 through 13 of 13 returned.

1. Document ID: US 6567084 B1

L5: Entry 1 of 13 File: USPT May 20, 2003

DOCUMENT-IDENTIFIER: US 6567084 B1
TITLE: Lighting effect computation circuit and method therefore

Detailed Description Text (26):
The memory 350 is included in the circuit 300 to store the results produced by the adder 340. The inclusion of the selection block 330 enables the second operand provided to the adder 340 to be selected from a plurality of potential operands based on operand selection information 332. The memory 350 may include a number of entries and require a number of address and control signals in order to provide the required data for a particular operation. Thus, the particular operation code being executed may include the addressing information (source address) required to access the memory 350.

Detailed Description Text (120):
The selection block 840 provides the second operand 814 to the first operation unit 810. The selection block 840 selects the second operand 814 from a set of potential operands based on selection information received. The selection information may be derived from the particular operation code executed, where the operation code may be determined from numerous operation codes that are pending for multiple threads. The set of potential operands from which the selection block selects the second operands 814 includes the first stored result in the first memory bypass register 820, the second stored result as stored in the second memory bypass register 760, and data stored in at least one of the first and second memories 830 and 870.

Full | Title | Citation | Front | Review | Classification | Date | Reference | Sequences | Attachments | Claims | KWIC | Draw Desc | Image

2. Document ID: US 6557098 B2

L5: Entry 2 of 13 File: USPT Apr 29, 2003

DOCUMENT-IDENTIFIER: US 6557098 B2
TITLE: Microprocessor including an efficient implementation of extreme value instructions

Abstract Text (1):
An execution unit is provided for executing a first instruction which includes an opcode field, a first operand field, and a second operand field. The execution unit includes a first input register for receiving a first operand specified by a value of the first operand field, and a second input register for receiving a second operand specified by a value of the second operand field. The execution unit further includes a comparator unit which is coupled to receive a value of the opcode field for the first instruction. The comparator unit is also coupled to receive the first and second operand values from the first and second input registers, respectively. The execution further includes a multiplexer which receives a plurality of inputs. These inputs include a first constant value, a second constant value, and the values of the first and second operand. If the decoded opcode value received by the comparator indicates

that the first instruction is either a compare or extreme value function, the comparator conveys one or more control signals to the multiplexer for the purpose of selecting an output of the multiplexer as the result of the first instruction. If the first instruction is one of a plurality of extreme value instructions, the one or more control signals conveyed by the comparator unit select between the first operand and second operand to determine the result of the first instruction. If the first instruction is one of a plurality of compare instructions, the one or more control signals conveyed by the comparator unit select between the first and second constant value to determine the result of the first instruction. In another embodiment, a similar execution unit is provided which handles vector operands.

Brief Summary Text (10):
The problems outlined above are in large part solved by an execution unit in accordance with the present invention. In one embodiment, an execution unit is provided for executing a first instruction which includes an opcode field, a first operand field, and a second operand field. The execution unit includes a first input register for receiving a first operand specified by a value of the first operand field, and a second input register for receiving a second operand specified by a value of the second operand field. The execution unit further includes a comparator unit which is coupled to receive a value of the opcode field for the first instruction. The comparator unit is also coupled to receive the first and second operand values from the first and second input registers, respectively. The execution further includes a multiplexer which receives a plurality of inputs. These inputs include a first constant value, a second constant value, and the values of the first and second operands. If the decoded opcode value received by the comparator indicates that the first instruction is either a compare or extreme value function, the comparator conveys one or more control signals to the multiplexer for the purpose of selecting an ouput of the multiplexer as the result of the first instruction. If the first instruction is one of a plurality of extreme value instructions, the one or more control signals conveyed by the comparator unit select between the first operand and second operand to determine the result of the first instruction. If the first instruction is one of a plurality of compare instructions, the one or more control signals conveyed by the comparator unit select between the first and second constant value to determine the result of the first instruction. In the case of the compare instructions, the value of the first and second constants may be advantageously chosen in order to form a mask for use by subsequent instructions. In another embodiment, a similar execution unit is provided which handles vector operands.

Detailed Description Text (52):
Turning now to FIG. 8, a block diagram of multimedia execution unit 36D is shown according to one embodiment of the invention. As depicted, execution unit 36D includes input registers 702A and 702B. Register 702A is coupled to floating point unit 36F by a first operand bus 710A. Likewise, register 702B is coupled to floating point unit 36F by a second operand bus 710B. Second operand bus 710B also couples input register 702B to receive operands via decode unit 20 and memory in one embodiment. Execution unit 36D further includes a comparator unit 730, which is coupled to receive decoded opcode values on decoded opcode bus 720 from decode unit 20. Comparator unit is further coupled to receive compare inputs (labeled "a" and "b") from register output buses 704A-B, coupled to the outputs of registers 702A and 702B, respectively. Comparator unit 730 conveys a select bus 732A as output.

Detailed Description Text (60):
As depicted, execution unit 37D includes input registers 701A and 701B. Register 701A is coupled to floating point unit 36F by a first operand bus 710A. Likewise, register 702B is coupled to floating point unit 36F by a second operand bus 710B. Second operand bus 710B also couples input register 701B to receive operands via decode unit 20 and memory in one embodiment. Unlike input registers 702 shown in FIG. 8, input register 701 are vectored. As shown, register 701A includes a first vector portion 703A and a second vector portion 703B. Likewise, register 701B includes a first vector portion 703C and a second vector portion 703D. Execution unit 37D further includes a comparator unit 731, which is similar to comparator unit 730 shown in FIG. 8 in that comparator unit 731 also receives a decode opcode value on bus 720. Comparator unit 731, however, is configured to perform two comparisons and convey two sets of select bus signals concurrently. Unit 731 receives a first set of signals (at inputs labeled "a.sub.1 " and "b.sub.1 ") which correspond to first vector portions 703A and 703C, as well as a second set of signals (at inputs labeled "a2" and "b2") which correspond to second vector portions 703B and 703D. In response to the comparison performed for inputs a.sub.1, and b.sub.1, comparator unit 731 conveys select bus 732A as output to multiplexer 740A. Similarly, comparator 731 conveys values on a select bus 732B to

multiplexer 740B in response to the comparison performed for the inputs labeled a.sub.2 and b.sub.2.

Full | Title | Citation | Front | Review | Classification | Date | Reference | Sequences | Attachments | Claims | KWIC | Draw Desc | Image

3. Document ID: US 6516407 B1

L5: Entry 3 of 13 File: USPT Feb 4, 2003

DOCUMENT-IDENTIFIER: US 6516407 B1
TITLE: Information processor

Detailed Description Text (13):
The second conditional instructions include operational instructions, transfer instructions and others except branch instructions. An ID code CRp in the instruction or a decoded signal thereof is provided to the control input of a selector 12C to select one of the second condition code registers of the second condition code register file 163, and the contents of the selected register is provided to a control circuit 123. The control circuit 123 compares the value of the condition flag S in the selected second condition code register and the reference flag SF in the instruction, and if the valid flag V in the selected second condition code register is `1,` and if S and SF both coincide with each other, then an operation indicated by an operation code OP is executed, or else the operation is not executed. RS1, RS2 and RD in the instruction are a first source operand, a second source operand and a destination operand, respectively, regarding the operation indicated by the operation code OP.

Detailed Description Text (70):
ID codes CRp and CRq in the instruction or decoded signals thereof are respectively provided to the control input of the selectors 12C and 12D to select two registers in the second code register file 163, and the outputs of the selectors are provided to the logic operation circuit 12E as RS1 and RS2. The logic operation circuit 12E performs one of operations shown in FIGS. 9 to 12 designated by the operation code OP in the instruction on the first source operand RS1 and the second source operand RS2, and the result thereof is provided to the demultiplexer 12B. An ID code CRk in the instruction or a decoded signal thereof is provided to the control input of the demultiplexer 12B to designate a second condition code resister in the second. condition code register file 163, and the operation's result is stored into the designated register.

Full | Title | Citation | Front | Review | Classification | Date | Reference | Sequences | Attachments | KWIC | Draw Desc | Image

4. Document ID: US 6438676 B1

L5: Entry 4 of 13 File: USPT Aug 20, 2002

DOCUMENT-IDENTIFIER: US 6438676 B1
TITLE: Distance controlled concatenation of selected portions of elements of packed data

Detailed Description Text (49):
The shift units 54a-d shift the bits they receive from the selection units 53a-d. In principle, each shift unit 53a-d provides for as many bits as the intermediate register 51. The amount of shift effected by each shift unit 53a-d is determined by the first operand, which is stored in the first intermediate register 50, and by the opcode. Each shift unit 54a-d may shift its bits by a different amount.

Detailed Description Text (52):

The functional unit is capable of performing various kinds of shift instructions, dependent on the opcode. For example, in response to an opcode that requires a normal shift on the content of the second operand as a whole, the instruction decoder may control the first selection unit 52a to select all of the bits from the second intermediate register 51, whereas the other shift units are controlled to select no bits at all; the shift amount may be passed from the first intermediate register 50 to the control input of the first shift unit 53a the first mask unit 54a receives a control signal to pass all bits from the first shift unit 53a to the OR unit 55; the other mask units 54b-c pass no bits. Similarly, in response to an opcode that requires separate shifts on individual numbers from different fields in the second operand: each selection unit 52a-d may be controlled to select the bits from a respective field of the second intermediate register 51 (e.g. a first and a second half of the second operand or a first, second, third and fourth quarter); shift amounts may be passed from respective fields in the first intermediate register 50 to the control inputs of respective shift units 53a-d; the mask units 54a,b receive control signals to pass respective fields of bits from the first and second shift unit 53a,b to the OR unit 55. More particularly, upon receiving a "bitconcat" instruction for a compression as shown in FIG. 4a, the instruction decoder will: cause the first and second selection unit 53a,b to select a first and second field from the second operand determine a shift amount by decrementing a length of the first field by a desired length of bits in the first field as determined from a length code in the first operand supply the shift amount as shift control signal to the second shift unit 54b and a zero shift amount to the first shift unit 54a cause the first and second mask unit 54a,b to pass only the bits of first field from both the first and the second shift unit 53a,b.

Full | Title | Citation | Front | Review | Classification | Date | Reference | Sequences | Attachments   KWIC | Draw Desc | Image

☐ 5. Document ID: US 6339821 B1

L5: Entry 5 of 13   File: USPT   Jan 15, 2002

DOCUMENT-IDENTIFIER: US 6339821 B1
TITLE: Data processor capable of handling an increased number of operation codes

Abstract Text (1):
A data processor is provided to increase the number of instructions it can handle, even with a large number of operands required for the instructions. The data processor comprises a decoding circuit (1) extracting bits (a1, a2) of an instruction as first operand fields and decoding an operation code, using the remaining bits (a4); an operand-field storage portion (3) including a first operand-field storage portion (3a) storing bits (a1, a2) obtained from the decoding circuit (1) via a selector (2), and a second operand-field storage portion (3b) storing a second operand field obtained on the basis of those bits (a2); and a data processing portion (5) receiving the first and the second operand fields from the operand-field storage portion (3) and processing data in registers designated by the first and the second operand fields.

Detailed Description Text (3):
FIG. 1 is a block diagram of a data processor according to a first preferred embodiment of the present invention. In FIG. 1, 1 is a decoding circuit (decoding portion) receiving an instruction IN of a program in order, extracting operand fields (first operand fields) of the instruction IN, and decoding an operation code of the instruction IN; 3 is an operand-field storage portion receiving the first operand fields from the decoding circuit 1 via a selector 2 and storing the first operand fields; 4 is a selector receiving operand fields D1, D2, D3, D4, . . . stored in the operand-field storage portion 3 and selecting three of them to output the selected operand fields as operand fields A1 to A3; 5 is a processing circuit (data processing portion) receiving the operand fields A1 to A3 and processing data in general registers designated by the operand fields A1 to A3; and 6 is a control circuit receiving a control signal S1 (decoding result of the operation code) from the decoding circuit 1 and on the basis of the control signal S1, generating and outputting control signals S2 to S5 for controlling the selector 2, the operand-field storage portion 3, the selector 4, and the processing circuit 5, respectively.

Detailed Description Text (10):
First, a latch circuit 11 latches the first instruction IN, "LD/#R0." From the mode bit "0" of the instruction, an instruction decoding portion 12 identifies the mode as the 8-bit mode. Then, the decoding circuit 1 extracts the eighth to fifteenth bits (a1, a2) of the instruction IN as first operand fields and decodes an operation code, using the remaining zeroth to seventh bits (a3, a4). The bit group a2 designates an address #R0. When the instruction decoding portion 12 judges the operation code as LD (data transfer instruction), the control circuit 6 controls the selector 2 and the operand-field storage portion 3 so that the selector 2 writes the bit group a2 into an operand-filed storage region 33 of the operand-field storage portion 3. That is, on the LD instruction, data previously prepared is written into a register designated by the operand-field storage region 33. When the control circuit 6 detects the completion of execution of the LD instruction, the operand-field storage portion 3 turns on a switch SW1 to transfer the operand field D3 in the operand-field storage region 33, out of the first operand fields, to a operand-field storage region 34 as a second operand field. Thus, the address #R0 is written to the operand-field storage region 34.

Detailed Description Text (11):
Next, the latch circuit 11 latches the next instruction IN, "ADDX/#R2, #R1." Since the mode bit of the instruction is "0", like the previous instruction, the decoding circuit 1 extracts the eighth to fifteenth bits (a1, a2) of the instruction IN as first operand fields and decodes an operation code, using the remaining zeroth to seventh bits (a3, a4). The bit groups a2 and a1 designate the addresses #R2 and #R1, respectively. When the instruction decoding portion 12 judges the operation code as ADDX, the control circuit 6 controls the selector 2 and the operand-field storage portion 3 so that the selector 2 writes the bit groups a1 and a2 into the operand-field storage regions 31 and 33 of the operand-field storage portion 3, respectively. At this time, the operand-field storage portions 34, 31, and 33 store the addresses #R0, #R1, and #R2, respectively. Then, the selector 4 outputs the operand field D1 in the operand-field storage region 31, the operand field D4 in the operand-field storage region 34, and the operand field D3 in the operand-field storage region 33 as the operand fields A1, A2, and A3, respectively. Accordingly, in the processing circuit 5, the selectors 52a and 52b read out data in the general register R1 and data in the general register R0, respectively; the arithmetic circuit 53 calculates a sum of the data in the general register R1 and the data in the general register R0; and the selector 54 writes the sum to the general register R2. When the control circuit 6 detects the completion of execution of the ADDX instruction, the operand-field storage portion 3 turns on the switch SW1 to transfer the operand field D3 in the operand-field storage region 33, out of the first operand fields, to the operand-field storage region 34 as a second operand field. Thus, the address #R2 is written to the operand-field storage region 34.

Detailed Description Text (16):
First, the latch circuit 11 latches the instruction IN, "ADD0/#R2, #R1, #R0." From the mode bit "1" of the instruction, the instruction detecting portion 12 identifies the mode as the 4-bit mode. Then, the decoding circuit 1 extracts the fourth to fifteenth bits (a1, a2, a3) of the instruction IN as first operand fields and decodes an operation code, using the remaining zeroth to third bits (a4). That is, in the 4-bit mode as compared in the 8-bit mode, the decoding circuit 1 increases the number of first operand fields. The bit groups a3, a2, and a1 designate the addresses #R2, #R1, and #R0, respectively. When the instruction decoding portion 12 judges the operation code as ADD0, the control circuit 6 controls the selector 2 and the operand-field storage portion 3 so that the selector 2 writes the bit groups a1, a2, a3 into the operand-field storage regions 31, 32, and 33 of the operand-field storage portion 3, respectively. Then, the selector 4 outputs the operand field D1 in the operand-field storage region 31, the operand field D2 in the operand-field storage region 32, and the operand field D3 in the operand-field storage region 33 as the operand fields A1, A2, and A3, respectively. Accordingly, in the processing circuit 5, the selectors 52a and 52b read out data from the general registers R0 and R1, respectively; the arithmetic circuit 53 calculates a sum of the data in the general register R0 and the data in the general register R1; and the selector 54 writes the sum into the general register R2.

Detailed Description Text (24):
First, the latch circuit 11 latches an instruction IN, "ADDY/#R2, #R0." Since the mode bit of the instruction is "0", the decoding circuit 1 extracts the eighth to fifteenth bits (a1, a2) of the instruction IN as first operand fields, and decodes an operation code, using the remaining zeroth to seventh bits (a3, a4). The bit groups a2 and a1 designate the addresses #R2 and #R0, respectively. When the instruction decoding portion 12 judges the operation code as ADDY, the control circuit 6 controls the selector 2 and the operand-field storage portion 3 so that the selector 2 writes the

bit groups a1 and a2 into the operand-field storage regions 31 and 33 of the operand-field storage portion 3, respectively. The bit group a1 is also inputted to the adder 7. The adder 7 writes data obtained by adding one to the bit group a1, namely, #R1, to the operand-field storage region 35. At this time, the addresses #R0, #R1, and #R2 are stored in the operand-field storage regions 31, 35, and 33, respectively. Then, the selector 4 outputs the operand filed D1 in the operand-field storage region 31, the operand field D5 in the operand-field storage region 35, and the operand field D3 in the operand-field storage region 33 as the operand fields A1, A2, and A3, respectively. Accordingly, in the processing circuit 5, the selectors 52a and 52b read out data from the general registers R0 and R1, respectively; the arithmetic circuit 53 calculates a sum of the data in the general register R0 and the data in the general register R1; and the selector 54 writes the sum into the general register R2.

Detailed Description Text (36):
First, the latch circuit 11 latches an instruction IN, "ST/#R0." From the mode bit "0" of the instruction, the instruction decoding portion 12 identifies the mode as the 8-bit mode. Then, the decoding circuit 1 extracts the eighth to fifteenth bits of the instruction IN (a1, a2) as first operand fields, and decodes an operation code, using the remaining zeroth to seventh bits (a3, a4). The bit group a2 designates the address #R0. When the instruction decoding portion 12 judges the operation code as ST, the control circuit 6 controls the selector 2 and the operand-field storage portion 3 so that the selector 2 writes the bit group a2 into the operand-field storage portion 31 of the operand-field storage portion 3. On the ST instruction, data stored in a register designated by the operand-field storage region 33 is written into a predetermined region. When the control circuit 6 detects the completion of execution of the ST instruction, the operand-field storage portion 3 turns on the switch SW2 to transfer the operand field D1 in the operand-field storage region 31, out of the first operand fields, to an operand-field storage region 36 as a second operand field. Thus, the address #R0 is written to the operand-field storage region 36.

Detailed Description Text (37):
Next, the latch circuit 11 latches the next instruction IN, "ADDZ/#R2, #R1." Since the mode bit of the instruction is "0", like the previous instruction, the decoding circuit 1 extracts the eighth to fifteenth bits (a1, a2) of the instruction IN as first operand fields, and decodes an operation code, using the remaining zeroth to seventh bits (a3, a4). The bit groups a2 and a1 designate the addresses #R2 and #R1, respectively. When the instruction decoding portion 12 judges the operation code as ADDZ, the control circuit 6 controls the selector 2 and the operand-field storage portion 3 so that the selector 2 writes the bit groups a1 and a2 into the operand-field storage regions 31 and 33 of the operand-field storage portion 3, respectively. At this time, the operand-field storage regions 36, 31, and 33 stores the addresses #R0, #R1, and #R2, respectively. Then, the selector 4 outputs the operand field Di in the operand-field storage region 31, the operand field D6 in the operand-field storage region 36, and the operand field D3 in the operand-field storage region 33 as operand fields A1, A2, and A3, respectively. Accordingly, in the processing circuit 5, the selectors 52a and 52b read data from the general registers R1 and R0, respectively; the arithmetic circuit 53 calculates a sum of the data in the general register R1 and the data in the general register R0; and the selector 54 writes the sum into the general register R2. When the control circuit 6 detects the completion of execution of the ADDZ instruction, the operand-field storage portion 3 turns on the switch SW2 to transfer the operand field D1 in the operand-field storage region 31, out of the first operand fields, to the operand-field storage region 36 as a second operand field. Thus, the address #R1 is written to the operand-field storage region 36.

Full | Title | Citation | Front | Review | Classification | Date | Reference | Sequences | Attachments   KWIC | Draw Desc | Image

☐ 6. Document ID: US 6223277 B1
L5: Entry 6 of 13 File: USPT Apr 24, 2001

DOCUMENT-IDENTIFIER: US 6223277 B1
TITLE: Data processing circuit with packed data structure capability

Detailed Description Text (24):
According to the preferred embodiment of the present invention, the second operand in the arithmetic or logical instruction may be an immediate operand, or the contents of one of the registers in register file 24. In this regard, bit position 24 (IO) in this instruction indicates, when set, that bit positions 23:16 contain an immediate operand value for use in the arithmetic or logical operation specified by the OPCODE portion of the instruction. If bit position (IO) is not set, bit positions 23:16 contain a second source register in register file 24, with bit positions 20:16 selecting one of the thirty-two registers in register file 24 as the source register, and bit positions 23:21 containing a three-bit code indicating the location (BYTEX, WORDx, or DWORD) within that source register, coded as described above. In this case of the second source register, the appropriate ones of operand multiplexers 26 and shift/mask units 28 again select the addressed portion of the addressed register in register file 24, for application to ALU 30 along with the first source operand determined by bit positions 15:8 as described above.

Full | Title | Citation | Front | Review | Classification | Date | Reference | Sequences | Attachments | KWIC | Draw Desc | Image

7. Document ID: US 6029244 A

L5: Entry 7 of 13 File: USPT Feb 22, 2000

DOCUMENT-IDENTIFIER: US 6029244 A
TITLE: Microprocessor including an efficient implementation of extreme value instructions

Abstract Text (1):
An execution unit is provided for executing a first instruction which includes an opcode field, a first operand field, and a second operand field. The execution unit includes a first input register for receiving a first operand specified by a value of the first operand field, and a second input register for receiving a second operand specified by a value of the second operand field. The execution unit further includes a comparator unit which is coupled to receive a value of the opcode field for the first instruction. The comparator unit is also coupled to receive the first and second operand values from the first and second input registers, respectively. The execution further includes a multiplexer which receives a plurality of inputs. These inputs include a first constant value, a second constant value, and the values of the first and second operand. If the decoded opcode value received by the comparator indicates that the first instruction is either a compare or extreme value function, the comparator conveys one or more control signals to the multiplexer for the purpose of selecting an output of the multiplexer as the result of the first instruction. If the first instruction is one of a plurality of extreme value instructions, the one or more control signals conveyed by the comparator unit select between the first operand and second operand to determine the result of the first instruction. If the first instruction is one of a plurality of compare instructions, the one or more control signals conveyed by the comparator unit select between the first and second constant value to determine the result of the first instruction. In another embodiment, a similar execution unit is provided which handles vector operands.

Brief Summary Text (10):
The problems outlined above are in large part solved by an execution unit in accordance with the present invention. In one embodiment, an execution unit is provided for executing a first instruction which includes an opcode field, a first operand field, and a second operand field. The execution unit includes a first input register for receiving a first operand specified by a value of the first operand field, and a second input register for receiving a second operand specified by a value of the second operand field. The execution unit further includes a comparator unit which is coupled to receive a value of the opcode field for the first instruction. The comparator unit is also coupled to receive the first and second operand values from the first and second input registers, respectively. The execution further includes a multiplexer which receives a plurality of inputs. These inputs include a first constant value, a second constant value, and the values of the first and second operands. If the decoded opcode value received by the comparator indicates that the first instruction is either

a compare or extreme value function, the comparator conveys one or more control signals to the multiplexer for the purpose of selecting an output of the multiplexer as the result of the first instruction. If the first instruction is one of a plurality of extreme value instructions, the one or more control signals conveyed by the comparator unit select between the first operand and second operand to determine the result of the first instruction. If the first instruction is one of a plurality of compare instructions, the one or more control signals conveyed by the comparator unit select between the first and second constant value to determine the result of the first instruction. In the case of the compare instructions, the value of the first and second constants may be advantageously chosen in order to form a mask for use by subsequent instructions. In another embodiment, a similar execution unit is provided which handles vector operands.

Detailed Description Text (48):
Turning now to FIG. 8, a block diagram of multimedia execution unit 36D is shown according to one embodiment of the invention. As depicted, execution unit 36D includes input registers 702A and 702B. Register 702A is coupled to floating point unit 36F by a first operand bus 710A. Likewise, register 702B is coupled to floating point unit 36F by a second operand bus 710B. Second operand bus 710B also couples input register 702B to receive operands via decode unit 20 and memory in one embodiment Execution unit 36D further includes a comparator unit 730, which is coupled to receive decoded opcode values on decoded opcode bus 720 from decode unit 20. Comparator unit is further coupled to receive compare inputs Gabeled "a" and "b") from register output buses 704A-B, coupled to the outputs of registers 702A and 702B, respectively. Comparator unit 730 conveys a select bus 732A as output.

Detailed Description Text (56):
As depicted, execution unit 37D includes input registers 701A and 701B. Register 701A is coupled to floating point unit 36F by a first operand bus 710A. Likewise, register 702B is coupled to floating point unit 36F by a second operand bus 710B. Second operand bus 710B also couples input register 701B to receive operands via decode unit 20 and memory in one embodiment. Unlike input registers 702 shown in FIG. 8, input register 701 are vectored. As shown, register 701A includes a first vector portion 703A and a second vector portion 703B. Likewise, register 701B includes a first vector portion 703C and a second vector portion 703D. Execution unit 37D further includes a comparator unit 731, which is similar to comparator unit 730 shown in FIG. 8 in that comparator unit 731 also receives a decode opcode value on bus 720. Comparator unit 731, however, is configured to perform two comparisons and convey two sets of select bus signals concurrently. Unit 731 receives a first set of signals (at inputs labeled "a," and "b,") which correspond to first vector portions 703A and 703C, as well as a second set of signals (at inputs labeled "a2" and "b2") which correspond to second vector portions 703B and 703D. In response to the comparison performed for inputs a.sub.1 and b.sub.1, comparator unit 731 conveys select bus 732A as output to multiplexer 740A. Similarly, comparator 731 conveys values on a select bus 732B to multiplexer 740B in response to the comparison performed for the inputs labeled a.sub.2 and b.sub.2.

Full | Title | Citation | Front | Review | Classification | Date | Reference | Sequences | Attachments | KWIC | Draw Desc | Image

---

☐ 8. Document ID: US 5574927 A

L5: Entry 8 of 13 File: USPT Nov 12, 1996

DOCUMENT-IDENTIFIER: US 5574927 A
TITLE: RISC architecture computer configured for emulation of the instruction set of a target computer

Detailed Description Text (3):
With reference now to the drawings, and particularly to FIGS. 1 and 2, there is shown in FIG. 2 a block diagram of a preferred embodiment of the present invention as added to a typical prior art RISC computer of FIG. 1. As shown in FIG. 1, core logic of a prior art RISC computer 10 is comprised of five main functional blocks, a microcode instruction memory 12, an instruction decoder 14, a register file 16, selection logic 18 and an arithmetic logic unit (ALU) 20. The prior art RISC computer 10 is constructed

to take microcoded instructions 22, stored in microcode instruction memory 12 and within a single clock cycle decode and process the majority of instructions. Within each instruction 22 there is found a plurality of fields, each comprised of a plurality of bits that primarily designate an opcode, a first operand, a second operand and a destination operand. An instruction decoder 14 decodes the opcode and selects operands from a register file 16 using selection logic 18 and directs the contents of the selected registers to the ALU 20. Additionally, the instruction decoder 14 selects the type of arithmetic instruction that the ALU 20 performs. The output of the ALU 20 is stored in the selected destination operand within the register file 16 as decoded by the instruction decoder 14 and selected by the selection logic 18. This RISC architecture is optimized for fast and efficient execution of microcoded instructions and results in a low cost chip with minimized power dissipation. However, this architecture is not particularly suitable for emulation of other computers.

Full | Title | Citation | Front | Review | Classification | Date | Reference | Sequences | Attachments | KWIC | Draw Desc | Image

☐ 9. Document ID: US 5345555 A

L5: Entry 9 of 13 File: USPT Sep 6, 1994

DOCUMENT-IDENTIFIER: US 5345555 A
TITLE: Image processor memory for expediting memory operations

CLAIMS:

5. A graphics memory system for effecting an algebraic logic operation between an operand of a given memory cell in said graphics memory system and at least one other operand and leaving the results thereof in the given memory cell with no more than one write to the given memory cell and without having to extract the operand therefrom, said graphics memory system comprising:

a memory cell array for storing binary data, cells of said memory cell array having memory cell modification means for receiving control instructions and performing logic operations according to the control instructions on memory cells of said memory cell array,

data storage selection means for addressing cells in said memory cell array,

raster opcode decode means for decoding a given raster opcode, first and second operands, and generating a control output accordingly, said raster opcode specifying said algebraic logic operation to be effected as required for implementing a predetermined raster operation, and said control output being representative of the control instruction required if any for effecting said algebraic logic operation between said first and second operands and the operand of the given memory cell as addressed by said data storage selection means so that the logic operation performed within the given memory cell is operative for modifying the operand therein so as to leave the results of said algebraic logic operation with the given memory cell, and

means for supplying the control output of said raster opcode decode means as the control instructions to the memory cell modification means of the given memory cell when the operational effect of said algebraic logic operation is dependent upon the operand of the given memory cell.

Full | Title | Citation | Front | Review | Classification | Date | Reference | Sequences | Attachments | KWIC | Draw Desc | Image

☐ 10. Document ID: US 4916606 A

L5: Entry 10 of 13 File: USPT Apr 10, 1990

DOCUMENT-IDENTIFIER: US 4916606 A
TITLE: Pipelined parallel data processing apparatus for directly transferring operand data between preceding and succeeding instructions

Detailed Description Text (26):
An exemplary circuit configuration of the arithmetic unit or accumulator 40 is shown in FIG. 6. Referring to the figure, the leftmost byte address (B.sub.1)+D.sub.1 of the real address register 26 is transferred to an A-register 430 through a line 26A. Additionally, a value equivalent to the number of the shifts performed by the aligner 36 for the first operand is loaded in a register 431 through a line 28A. The operation code OP available from the register 23 through a line 23A is supplied to the input of a micro-instruction controller 400. The first and second operand data outputted from the aligners 36 and 37 are, respectively, supplied to selectors 490 and 491 provided according to another aspect of the invention through lines 36A and 37A. The selector 490 serves to select the line 36A for the first operand data when both lines 71A and 73A are at logic "0", while selecting a data line 481A of a W2-register 481 when the lines 71A and 73A are both at logic "0" level. On the other hand, in case the lines 71A and 73A are either logic "1" and "0" or "1" and "1", respectively, a data line 480A of a W1-register 480 is selected. The selected data is then outputted on a line 490A. In a similar manner, the selector 491 serves to select a line 37A for the second operand data when lines 70A and 72A are, respectively, logic "0" while selecting a data line 481A of a W2-register 481 when the states of the lines 70A and 72A are, respectively, logic "0" and "1". On the other hand, in case the line 70A and 72A are either logic "1" and "0" or "1" and "1", respectively, the data line 480A of the W1-register 480 is selected. The selected data is then outputted on a line 491A. The data on the line 490A and 491A are placed in an X-register 410 and a Y-register 420, respectively, in synchronism with the stage L. The significance of the selector mentioned above will be described below.

Full | Title | Citation | Front | Review | Classification | Date | Reference | Sequences | Attachments KWIC | Draw Desc | Image

☐ 11. Document ID: US 4761754 A

L5: Entry 11 of 13 File: USPT Aug 2, 1988

DOCUMENT-IDENTIFIER: US 4761754 A
TITLE: Vector processor wherein outputs of vector registers are fixedly coupled to inputs of vector calculators

Detailed Description Text (7):
When the vector instruction is an operation instruction indicative of the vector addition by the operation code OP, the output selecting circuit 37 selects the first vector operation unit 31 as the acting operation unit. The first operand R1 specifies a destination register. The second and the third operands R2 and R3 specify source registers. Responsive to the second and the third operands R2 and R3, the output selecting circuit 37 delivers the vectors held in the source registers to the respective input terminals of the first vector operation unit 31 as the operand vectors. Responsive to the first operand R1, the input selecting circuit 36 loads the destination register with the result vector produced at the output terminal of the first vector operation unit 31.

Detailed Description Text (13):
When the vector instruction is a store instruction in which the operation code OP indicates a store operation, the store selecting circuit 49 selects that one of the vector registers 21 through 28 as a source register which is specified by the second operand R2. The store selecting circuit 49 transfers the vector held in the source register to the store register 38 (FIG. 3) for storage thence in that one of the memory addresses of the main memory 11 stored in the address register which is specified by

the first operand R1 as a destination address.

Full | Title | Citation | Front | Review | Classification | Date | Reference | Sequences | Attachments | KWIC | Draw Desc | Image

12. Document ID: US 4754424 A

L5: Entry 12 of 13 File: USPT Jun 28, 1988

DOCUMENT-IDENTIFIER: US 4754424 A
TITLE: Information processing unit having data generating means for generating immediate data

Detailed Description Text (2):
Referring to FIG. 1, a preferred embodiment of the invention comprises a main memory unit 100 for storing instruction words (IW's) and data, and a processing unit 200. The unit 200 further comprises a main memory control circuit 201, a 32-bit instruction register 202, a numerical data generator circuit 205, a logical data generator circuit 206, a register file 203 having 128 64-bit arithmetic registers, a decoder 204, an arithmetic circuit 207 and selector circuits 208 to 211. The instruction register 202 functions to temporarily store one IW read out of the unit 100 via the circuit 201. Bit positions 0 to 7, bit positions 8 to 15, bit positions 16 to 23 and bit positions 24 to 31 of an IW stored in the register 202 constitute an operation code OP for designating the type of instruction to be executed, a register designating field R for designating one of the arithmetic registers in the register file 203, a first operand designating field S1 for designating first immediate data or one of the arithmetic registers, and a second operand designating field S2 for designating second immediate data or another one of the arithmetic registers. The field OP of the IW is given from the instruction register 202 to the decoder 204 to generate various control signals required for execution of the instruction. The circuit 205, as will be described in further detail below, functions to generate desired 64-bit integral data in response to the field S1 of the IW. The circuit 206, as will also be described in further detail below, functions to generate desired 64-bit logical data in response to the field S2 of the IW. The circuit 210 selects either data supplied from the circuit 205 or those supplied from the register file 203 in response to a control signal from the decoder 204. The circuit 211 selects either data supplied from the circuit 206 or those supplied from the register file 203 in response to the control signal of the decoder 204. The arithmetic circuit 207 processes data supplied from the circuit 210 and/or the circuit 211 as demanded by the field OP of the IW. Data supplied from the arithmetic circuit 207 is either stored in the memory unit 100 via the circuit 201 or given to the circuit 208. The circuit 208 selects either the data supplied from the arithmetic unit 207 or those read out of the memory unit 100 via the circuit 201 to supply the selected data to the register file 203. To designate one of the arithmetic registers in the register file 203, the circuit 209 selects one of the field R, S1 and S2 of the IW stored in the register 202. The circuit 201 has an instruction buffer comprising, for instance, two 64-bit instruction buffer registers A and B.

Full | Title | Citation | Front | Review | Classification | Date | Reference | Sequences | Attachments | KWIC | Draw Desc | Image

13. Document ID: US 3764988 A

L5: Entry 13 of 13 File: USPT Oct 9, 1973

DOCUMENT-IDENTIFIER: US 3764988 A
TITLE: INSTRUCTION PROCESSING DEVICE USING ADVANCED CONTROL SYSTEM

Detailed Description Text (37):

The references 26A, 26B, 26C and 26D denote transfer lines for transferring instructions from the memory unit to the instruction buffer registers 5A, 5B, 5C and 5D respectively, 27A, 27B, 27C and 27D transfer lines for transferring instructions to the instruction register 13 from the instruction buffer registers 5A, 5B, 5C and 5D respectively, 28A, 28B, 28C and 28D selection lines for selecting the transfer line 27A, 27B, 27C or 27D according to the indication from the selection circuit 12, 29 a transfer line for transferring the operation code 13A of the instruction register 13 to the decoder circuit 14, 31 an output line carrying an output when the decoder 14 determines that the presented instruction is not a branch instruction, 32 an output line carrying an output when the decoder determines that the presented instruction is a branch instruction, 33 a transfer line for transferring the operand address field 13C of the instruction register 13 to the register 15, 34 a transfer line for transferring the index register number 13B of the instruction register to the index register group (not shown diagrammatically), 35 a transfer line for transferring the operand address of the first instruction register 6 (FIG. 3), 36 a transfer line for transferring the address of the index register designated by the index register number 13B, 37 a signal line indicating that the condition of the branch instruction has been established, 38 an output line for setting the flip-flop 17, 39 an output line of the counter 11, 40 and 41 output lines of the registers 15 and 16 respectively, and 42 a transfer line for transferring the address of the index register selected according to the index register number of the first instruction register 6 (FIG. 3).

Full | Title | Citation | Front | Review | Classification | Date | Reference | Sequences | Attachments | KWIC | Draw Desc | Image

Generate Collection | Print

Terms	Documents
L4 same (operation code or opcode)	13

Display Format: KWIC | Change Format

Previous Page   Next Page